This listing of claims will replace all prior versions, and listings, of claims in the application.

LISTING OF CLAIMS:

Claim 1 (Currently Amended): A method for characterizing runtime behavior of a computer program executing in an execution environment, said method comprising:

a) inserting yield points comprising code to be executed at distinguished locations of a program to be executed, each said yield point indicating a ~~potential~~ conditional sampling operation during execution of said program;

b) during program execution, ~~identifying~~ unconditionally executing a yield point instance and, in response to ~~an identified~~ executing said yield point instance, ascertaining a state of said execution environment for indicating whether [[a]] the conditional sampling operation is to be performed; and,

c) when a state of said execution environment indicates a condition for performing said sampling operation, recording relevant information for characterizing behavior of said execution environment, whereby conditional sampling operations performed at unconditionally executed yield points occur at a subset of the executions of yield points.

Claim 2 (Original): The method as claimed in Claim 1, wherein said sampling operation includes identifying a method currently executing in said program, said method including tracking frequencies of methods executed in said program for characterizing said program behavior.

Claim 3 (Original): The method as claimed in Claim 2, wherein said sampling operation includes identifying a calling context associated with methods called by said program, said method including tracking calling context frequency for characterizing said program behavior.

Claim 4 (Original): The method as claimed in Claim 1, wherein said sampling operation includes identifying current program variable values, said program variable values being tracked for characterizing said program behavior.

Claim 5 (Original): The method as claimed in Claim 1, wherein said sampling operation includes identifying basic blocks executed in said program, said method including tracking a frequency of basic blocks for characterizing said program behavior.

Claim 6 (Original): The method as claimed in Claim 1, wherein when said state of said execution environment does not indicate a sampling operation, the step of executing a next instruction in said executing program after said identified yield point.

Claim 7 (Original): The method as claimed in Claim 1, wherein said step b) of ascertaining a state of said execution environment includes checking status of a trigger bit set by said execution environment to indicate performance of said sampling operation.

Claim 8 (Original): The method as claimed in Claim 1, wherein said trigger bit status is set periodically by said executing environment.

Claim 9 (Original): The method as claimed in Claim 8, further including the steps of:

invoking a runtime system interrupt at periodic time intervals; and,

implementing an interrupt handler mechanism for catching said interrupt and setting said trigger bit.

Claim 10 (Original): The method as claimed in Claim 2, wherein said step of identifying a currently executing method comprises determining an instruction address at which the yield point was taken and mapping that address to a called method.

Claim 11 (Original): The method as claimed in Claim 3, wherein said step of identifying a calling context associated with methods comprises inspecting a call-stack runtime data structure for tracking methods currently active in said executing program.

Claim 12 (Original): The method as claimed in Claim 1, further including the step of implementing a compiler device for inserting one or more yield points in said program.

Claim 13 (Original): The method as claimed in Claim 1, further including the step of implementing an interpreter device for ensuring execution of said yield points in said program.

Claim 14 (Original): The method as claimed in Claim 1, wherein said yield points are inserted in one or more program locations including: a method prologue and a loop back edge.

Claim 15 (Currently Amended): A method for characterizing runtime behavior of a computer program executing in an execution environment, said method comprising:

a) inserting yield points comprising code to be executed at distinguished locations of a program to be executed, each said yield point indicating a ~~potential~~ conditional sampling operation during execution of said program;

b) during program execution, ~~identifying~~ unconditionally executing a yield point instance;

c) counting a number of ~~identified~~ executed yield points;

d) comparing said number against a predetermined threshold; and,

e) in response to meeting said threshold, performing a sampling operation of said executing program, and, recording relevant information for characterizing behavior of said execution environment in response to said sampling, whereby conditional sampling operations performed at unconditionally executed yield points occur at a subset of the executions of yield points.

Claim 16 (Original): The method as claimed in Claim 15, wherein said sampling operation includes identifying a method currently executing in said program, said method including tracking frequencies of methods executed in said program for characterizing said program behavior.

Claim 17 (Original): The method as claimed in Claim 16, wherein said sampling operation includes identifying a calling context associated with methods called by said program, said method including tracking calling context frequency for characterizing said program behavior.

Claim 18 (Original): The method as claimed in Claim 15, wherein said sampling operation includes identifying current program variable values, said program variable values being tracked for characterizing said program behavior.

Claim 19 (Original): The method as claimed in Claim 15, wherein said sampling operation includes identifying basic blocks executed in said program, said method including tracking a frequency of basic blocks for characterizing said program behavior.

Claim 20 (Original): The method as claimed in Claim 15, wherein said step c) includes the steps of:

initializing a counter to said predetermined threshold; and, for each identified yield point instance,

decrementing said counter until said counter is zero, whereby said sampling operation is arranged such that a fixed percentage of all executed yield points are taken.

Claim 21 (Original): The method as claimed in Claim 16, wherein said step of identifying a currently executed method comprises determining an instruction address at which the yield point was taken and mapping that address to a called method.

Claim 22 (Original): The method as claimed in Claim 17, wherein said step of identifying a calling context associated with methods comprises inspecting a call-stack runtime data structure for tracking methods currently active in said executing program.

Claim 23 (Original): The method as claimed in Claim 15, further including the step of implementing a compiler device for inserting one or more yield points in said program, said yield points being in one or more program locations including: a method prologue and a loop back edge.

Claim 24 (Original): The method as claimed in Claim 15, further including the step of implementing an interpreter device for ensuring execution of said yield points in said program.

Claim 25 (Currently Amended): A system for characterizing runtime behavior of a computer program executing in an execution environment, said system comprising:

a) means for inserting yield points comprising code to be executed at distinguished locations of a program to be executed, each said yield point indicating a ~~potential~~ conditional sampling operation during execution of said program;

b) mechanism for ~~identifying~~ unconditionally executing instances of yield points inserted in said executing program;

c) control device for determining a condition for performing a sampling operation of said executing program at an ~~identified~~ executed yield point instance; and,

d) sampling device for performing said sampling operation of said executing program upon satisfaction of said condition, and recording relevant information for characterizing behavior of said execution environment in response to said sampling, whereby conditional sampling operations performed at unconditionally executed yield points occur at a subset of the executions of yield points.

Claim 26 (Original): The system as claimed in Claim 25, wherein said sampling device includes mechanism for identifying a method currently executing in said program; said sampling device comprising mechanism for tracking frequencies of methods executed in said program for characterizing said program behavior.

Claim 27 (Original): The system as claimed in Claim 26, wherein said sampling device includes mechanism for identifying a calling context associated with methods called by said program, said tracking mechanism further tracking calling context frequency for characterizing said program behavior.

Claim 28 (Original): The system as claimed in Claim 25, wherein said sampling operation includes mechanism for identifying current program variable values, said tracking mechanism further tracking said program variable values for characterizing said program behavior.

Claim 29 (Original): The system as claimed in Claim 25, wherein said sampling device includes mechanism for identifying basic blocks executed in said program, said tracking mechanism further tracking a frequency of basic blocks for characterizing said program behavior.

Claim 30 (Original): The system as claimed in Claim 25, further including:

a system location for storing a trigger bit; and,

a runtime system for said executing environment, said runtime system setting said trigger bit to indicate performance of said sampling operation; wherein, said control device ascertains a state of said system bit for determining said sampling condition.

Claim 31 (Original): The system as claimed in Claim 30, wherein said runtime system includes:

interrupt mechanism for generating timer interrupt signal; and,

interrupt handler mechanism for catching said interrupt and setting said trigger bit.

Claim 32 (Original): The system as claimed in Claim 26, wherein said mechanism for identifying a currently executed method comprises includes determining an instruction address at which the yield point was taken and mapping that address to a called method.

Claim 33 (Original): The system as claimed in Claim 27, wherein said mechanism for identifying a calling context associated with methods comprises inspecting a call-stack runtime data structure for tracking methods currently active in said executing program.

Claim 34 (Original): The system as claimed in Claim 25, wherein said control device comprises:

counter device for counting a number of identified yield points; and,

device for comparing said number against a predetermined threshold value, wherein, in response to meeting of said threshold, said control device initiating performing of said sampling operation.

Claim 35 (Original): The system as claimed in Claim 25, further including a compiler device for inserting one or more yield points in said program.

Claim 36 (Original): The system as claimed in Claim 25, further including an interpreter device for ensuring execution of said yield points in said program.